MISB Motion Imagery Standards Board **STANDARD** **KLV Metadata in Motion Imagery**	**MISB ST 0107.3** **1 November 2018**

1 Scope

This Standard defines the baseline requirements for all implementations of KLV within Motion Imagery files and streams. It applies retroactively to all documents approved by the Motion Imagery Standards Board (MISB).

2 References

[1] SMPTE ST 336:2017 Data Encoding Protocol Using Key-Length-Value.
[2] MISB MISP-2019.1: Motion Imagery Handbook, Nov 2018.
[3] MISB ST 0601.14 UAS Datalink Local Set, Nov 2018.
[4] ITU Rec X.680 (08/15) Abstract Syntax Notation One (ASN.1): Specification of basic notation.
[5] IEEE 754-2008 Standard for Floating-Point Arithmetic [and Floating-Point formats].
[6] MISB ST 1201.3 Floating Point to Integer Mapping, Oct 2017.
[7] ISO/IEC 10646:2014 Information technology – Universal Coded Character Set (UCS).

3 Acronyms

BER	Basic Encoding Rules
KLV	Key Length Value
MISB	Motion Imagery Standards Board
MISP	Motion Imagery Standards Profile
OID	Object IDentifer
SMPTE	Society of Motion Pictures and Television Engineers

4 Modifications and Changes

Revision	Date	Summary of Changes
0107.3	11/01/2018	• Expanded document to include KLV requirements applicable to all MISB documents • Changed Title to reflect expanded scope

		• Added requirements -03 through -13

5 Introduction

This Standard defines a baseline set of requirements applicable to all MISB KLV metadata standards which promote interoperability amongst Motion Imagery systems, minimize bandwidth in transmission, and lessen the storage requirements of Motion Imagery.

6 Baseline Requirements

6.1 Bit and Byte Order

For systems to understand each other's data representations it is essential all metadata use the same bit and byte order convention in the storage of the data. This does not directly apply to data within a communication system, since these systems may change data order along a transmission path. The assumption is a communications system delivers data with the same bit or byte order in which it was received.

Computing systems order multiple bytes of data with a specific "endian-ness". The "big-endian" format in this Standard signifies the "big end" (most significant bit or byte value in the sequence) is stored first. It is synonymous with the terms most significant bit (msb) first, or most significant byte (MSB) first.

Requirement(s)	
ST 0107.2-01	Bit order shall be big-endian or msb.
ST 0107.2-02	Byte order shall be big-endian or MSB.

6.2 SMPTE KLV

The transmission of Motion Imagery data is typically limited by a fixed-bandwidth channel, so reducing any overhead in the metadata allows more capacity for the Motion Imagery content. KLV developed by the SMPTE is a bit-efficient data encoding format for representing metadata (see [1]). The MISB standards define requirements which incur limits on the full SMPTE 336, making the MISB standards a profile of the SMPTE standards. The Motion Imagery Handbook [2] discusses various KLV constructs in use throughout MISB standards.

Requirement	
ST 0107.3-03	All Local Set KLV metadata shall be formatted in compliance with SMPTE ST 336 [1].

6.3 KLV Local Sets

The most common construct in the MISB Standards is the KLV Local Set. A Local Set is a KLV construct for encapsulating a list of Local Set items as a single block of data (in bytes). A Local Set item is a Key-Length-Value triplet. The "Key" in a Local Set is represented as a "Tag", which only has meaning within the bounds of the Local Set in which it is defined. The Tag indicates which item, of the list of items, the Value represents. It provides the identity of the item, and via its defining document, provides the method to decode the Value into a software data type. The Length defines the number of bytes used by the Value. Lengths are usually positive numbers; however, a zero length is possible in unique cases. Values are the binary representation describing a specific item. In the case of a zero Length, the Value is not a part of the item (see MISB ST 0601 [3]). Please refer to the Motion Imagery Handbook for more details on Local Sets.

One advantage of KLV is the ability to skip or ignore items which may not be known by a consumer of the data. For example, suppose a new version of a Local Set introduces new tags which an existing decoder does not understand. Instead of crashing, the decoder simply ignores any item it does not understand – this makes a decoder "future-proof". In such cases, the Length specified by the Tag is used to skip the correct number of bytes to the next item.

Requirement	
ST 0107.3-04	Applications which decode MISB KLV Local Sets shall skip unknown Local Set values so as to not impact the decoding of known Local Set items within the same Local Set instance.

6.3.1 Tag Encoding

The Basic Encoding Rules (BER) for an Object Identifier (BER-OID) as defined in ASN.1 [4] is the method selected for encoding a Local Set tag. The use of BER-OID for tags provides a means to specify a tag number without the need to indicate the quantity of bytes used for the tag (i.e. the tag-length). BER-OID tags are one or more bytes linked together in chain fashion. In BER-OID a zero in the Most Significant Bit (msb) position terminates the tag's chain. For example, when the msb is set to zero in the first byte, this forms a one-byte tag number ranging from 0x00 to 0x7F. When the msb of the first byte is set to one, the second successive byte becomes part of the BER-OID chain. In continued fashion, if the second byte's msb is set to one, the next byte becomes part of the chain. This pattern continues until the msb of a final byte in the chain is zero. Together the seven Least Signification Bits (lsb) of each byte in the chain form the tag number. The Motion Imagery Handbook provides further details on BER. Using BER-OID for the tags in all MISB local sets enables efficient tag encoding and unlimited future growth.

To prevent BER-OID from including leading zeros, ASN.1 forbids the use of 0x80 in the first byte of a BER-OID value.

6.3.2 Length Encoding

The Length of a Local Set item uses BER for both short and long form encoding of octets. This encoding method provides the greatest degree of flexibility for variable length data contained within a KLV packet. BER is an efficient encoding of a length; however, by either using the long form for values less than 128, or by prepending a long form value with zero-byte values, BER becomes less efficient. For example, encoding the length of two (2), a value less than 128, with long form uses two bytes (0x8102) instead of the short form one-byte (0x02) length. Another example is encoding the value 128 with padded zeros (0x8300 0080) instead of the optimized value with two bytes (0x8180). The Motion Imagery Handbook provides information and illustrations for implementing the short and long forms of BER encoding.

Requirement	
ST 0107.3-05	All instances of item Length fields within a MISB defined KLV Universal or Local Set shall be BER Short form or BER Long form encoded using the fewest possible bytes.

6.3.3 Value Encoding

When encoding individual numeric (i.e. floats and integers) or character items within a Local Set, use the fewest number of bytes allowed. Local Set items may have a required or variable "value-length." Items with required value-lengths use the exact number of bytes specified by the required Length. Items with variable value-length will adjust in size based on the Value and the items may have maximum lengths defined. When encoding a Value do not exceed the maximum limits. Reduce variable length items to their smallest length without losing their meaning or precision.

Requirement(s)	
ST 0107.3-06	Where a MISB standard defines a numeric required Length for a Local Set item's Value, the encoded Value shall match the number of bytes defined by the required Length.
ST 0107.3-07	Where a MISB standard defines a numeric maximum Length for a Local Set item's Value, the encoded Value shall not exceed the number of bytes defined by the maximum Length.

6.3.3.1 Numeric Items

Numeric Items consist of Floating-Point Values, IMAP Floating Point Values, Signed Integer, Unsigned Integers, BER-OID, and BER Values.

Items using the "float", "float32", "float64" types in the MISB Standards will use the IEEE 754-2008 [5] standard to encode their values. The IEEE 754-2008 binary32 (4 bytes) and binary64 (8 bytes) are the two types of Floating-Point Values in use by the MISB Standards. However, other types of IEEE 754-2008 are possible in future updates.

Items using the "IMAPA" or "IMAPB" types in the MISB Standards will use the methods in MISB ST 1201 [6] to encode (or map) floating point values into an integer-based format. The IMAP method save bandwidth on individual floating-point values by converting a floating-point value into an integer using less bytes than the original floating-point value. The IMAP algorithm uses the allowed value range (i.e. minimum and maximum values) and the desired precision to map the value. With some values the precision is based on system requirements and can change from system to system, or even packet to packet. To use IMAP efficiently it is important to use the fewest number of bytes while maintaining the desired precision of the system.

Items using the “int” or “uint” prefix (e.g. int8, int16, uint8, etc.) types in the MISB Standards will use big-endian integers to encode their values. Some “int” or “uint” values define variable lengths that adjust based on their numeric value. With the variable length “int” or “uint” values, use the fewest bytes to represent the Value; do not include leading zero (or sign extensions) bytes.

Items using “ber-oid” types in the MISB standards use the ASN.1 [4] Basic Encoding Rules for Object Identification (BER-OID) rules. The Motion Imagery Handbook discusses the BER-OID encoding technique in detail with examples and figures. This technique adjusts the length based on the numeric value and the rules prevent wasted bytes in the encoding.

Items using “ber” types in the MISB standards use the ASN.1 [4] Basic Encoding Rules for Lengths rules. The Motion Imagery Handbook discusses the BER encoding technique in detail with examples and figures. Section 6.3.2 above explains the possible inefficiencies that can occur when using the BER technique.

Requirement(s)	
ST 0107.3-08	Where a MISB standard defines a Local Set item with a KLV format of, or prefixed with, “float,” the encoded Value shall use the IEEE 754-2008 [5] standard to encode the value.
ST 0107.3-09	Where a MISB standard defines a Local Set item with a KLV format of “IMAPA” or “IMAPB,” and the Local Set item allows variable length encoding, the encoded Value shall use the fewest number of bytes, while maintaining system precision.
ST 0107.3-10	Where a MISB standard defines a Local Set item with a KLV format of, or prefixed with, “uint” or “int,” and the Local Set item allows variable length encoding, the encoded value shall remove leading zero or leading sign extension bytes.
ST 0107.3-11	Where a MISB standard defines a Local Set item with a KLV format of “ber” and the item allows variable length encoding, the encoded Value shall use the fewest number of bytes.

6.3.3.2 String Items

Items with the type “utf8” are strings of characters based on the ISO/IEC 10646:2014 Universal Character Set (UCS) [7] standard for interpretation, as discussed in the Motion Imagery Handbook. Unless a Local Set item description states otherwise, all strings are variable length. The Local Set item’s length determines the length of the string with the one exception of the “empty string”.

The empty string uses a single null character (0x00) with no other characters to indicate the string is blank. The length of the empty string within KLV is one (1), for the null character, but the decoding software converts the empty string to the software language’s equivalent of an empty string (e.g. Java interprets an empty string as a String of length zero). The empty string is different than an “Unknown” string (see MISB ST 0601 [3]), where the string is undefined (e.g.

Java interprets an Unknown string as a String variable set to “null”). A string item with a length of zero (0) signifies an “Unknown” string.

Strings can easily consume a large amount of bandwidth; the recommended practice is to use the fewest characters needed to provide the intended meaning. With exception of the single null character for an empty string, or if an item definition allows it, do not use the characters in the hex ranges (inclusive) of 0x00 through 0x08, 0x0B through 0x0C, 0x0E through 0x1F, and character 0x7F. Furthermore, remove any leading or trailing null, tab, line feed, carriage return, and space characters (0x00, 0x09, 0x0A, 0x0D, and 0x20, respectively) in the Value unless specified by the defining document.

Requirement(s)	
ST 0107.3-12	Where a MISB standard defines a Local Set item with a KLV format of “utf8” and the Value is not the empty string, the encoded value shall have all leading and trailing, null (0x00), tab (0x09), line feed (0x0A), carriage return (0x0D), and space (0x20) characters removed, unless the item definition allows them.
ST 0107.3-13	Where a MISB standard defines a Local Set item with a KLV format of “utf8” and the Value is not the empty string, the encoded value shall have all characters in the hex ranges (inclusive) of 0x00 through 0x08, 0x0B, 0x0C, 0x0E through 0x1F, and character 0x7F removed, unless the item definition allows them.